Official URL

DOI : https://doi.org/10.1007/978-3-319-92997-2_9

SysML Models Verification and Validation in an Industrial Context: Challenges and Experimentation

Ronan Baduel[1,2](✉), Mohammad Chami[1], Jean-Michel Bruel[1], and Iulian Ober[1]

[1] IRIT, University of Toulouse, Toulouse, France
ronan.baduel@irit.fr

[2] Bombardier Transportation, Crespin, France

Abstract. This paper presents a solution for SysML model verification and validation, with a return of experience from its implementation in an industrial context. We present this solution as a way to overcome issues regarding the use of SysML in an industrial context. We contribute by providing a method and a list of the existing challenges and experimenta-tion results. We advocate the need to have semantics for SysML models without having to define a full domain-specific modeling language. We highlight the work, requirements and benefits that arise from the appli-cation of existing technical solutions, and hint at new perspectives and future development in system verification and validation.

Keywords: MBSE · Model verification · Systems modeling · SysML

1 Introduction

This paper focuses on the *verification* and *validation* (V&V) of system models, built as part of the system development process at Bombardier Transportation (BT) for producing a broad portfolio of railway products. The Systems Modeling Language (SysML) [1] is used to develop the system models based on a BT customized System Modeling Method (SysMM) [2]. The main objective is to develop a generic V&V solution based on SysML without any tool dependent criteria so that it is reusable across all BT divisions and projects.

1.1 MBSE and V&V

Systems engineering is an interdisciplinary process for supporting the system life cycle. Model-Based Systems Engineering (MBSE) introduces new capability into systems engineering practice and is defined by INCOSE as *"the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases"* [3].

https://doi.org/10.1007/978-3-319-92997-2_9

As there are several definitions of the terms *verification* and *validation* (V&V), we refer in this paper to the definitions of the standard IEEE 1012-2012 [4], which we apply in the context of model development, and not on the scale of the whole development process. From this point of view, *verification* ensures that the models created during the early steps of the development process have been correctly built, meaning they are free of errors and represent a coherent system. As for *validation*, it ensures that the system represented by the models match the requirements traced to the information displayed or induced.

1.2 Motivating Example

From a technical point of view, the main aspect in verifying a model is ensuring that no errors were made in the specification of the system design. Creating models and having correct models are two different things, and can impact the rest of the development process. Similarly to validation, the earlier an error is detected, the less the cost [5].

From an organizational point of view, within large organizations, ensuring that everyone create models under the same guidelines and constraints is a challenging task. It is crucial that the modeling team members work the same way and are able to exchange and communicate around their delivered models with others without any misunderstanding or consistency issues. This gets more complex with teams spread across continents and/or companies. Having one defined modeling method across an organization and applying it the same way are two different things.

During the early phases of MBSE adoption at BT, the focus on models' V&V was triggered mainly by specific projects based on particular customers or countries needs. As MBSE enabled the reuse of models specification across projects, the goals of V&V was extended towards being more generic and project-independent. This however introduced the discovery of hundreds of errors by the BT V&V team even sometimes for a single verification or validation rule. It was not that the models were globally false, but rather that the project specific teams had their own interpretation of the method or specific modeling practice, gained from experience. What it did mean is that the models could neither be easily reused by other teams, nor could they be adapted while reproducing the same modeling approach. This is a main challenge for large organizations that are driven by project specific customers in contrast with those able to generalize their products and offer a predefined product portfolio (e.g., in automotive). Therefore, the need for reusing V&V of the delivered models is crucial to ensure proper systems models reuse. Moreover, it is crucial to implement the suitable adoption approach, similar to the D3 MBSE Adoption Toolbox [6].

1.3 Outline of the Paper

Section 2 presents BT SysMM, how it was implemented, and what were the specifications of the verification solution to be developed. Section 3 discusses the state of the art introducing the solutions on which we based our work on.

Section 4 presents our solution and the work realized, providing a return on experience. Section 5 shows an example of application, and Sect. 6 describes the challenges known beforehand as well as those encountered when developing and implementing our solution. Finally, Sect. 7 concludes on what has been done and gives future directions for this work.

2 Background on BT SysMM

MBSE has been deployed at BT for several years across various applications, for instance in requirements, functional and safety engineering [2]. The BT System Modeling Method (SysMM) [2] consists of three main tasks. Each one aims at analyzing the system of interest (SOI) on a specific abstraction level (see Fig. 1). SysMM describes how BT engineers analyze, define, and represent their SOI using system models. The purpose of SysMM is to manage complexity and increase quality of the design artifacts to reduce development costs.

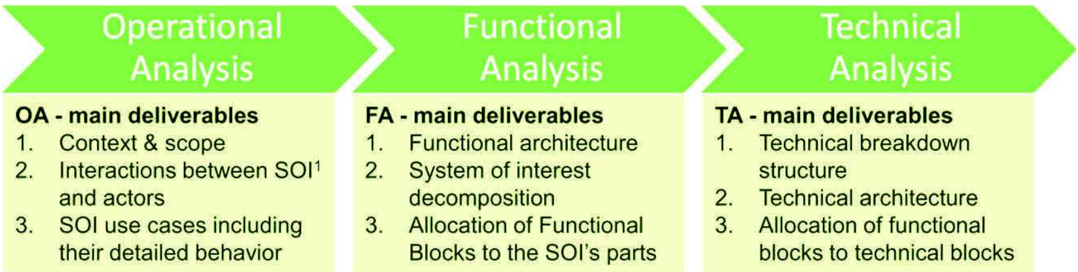

Fig. 1. BT system modeling method tasks [2]

The need for V&V of the SysMM system models was addressed from the beginning. SysMM tasks include V&V activities to ensure the quality of the deliverables. For instance, the *Operational Analysis* deliverables (left side of Fig. 1) are first verified automatically by the system modeling tool to check if the model elements and diagrams are conform with the associated guidelines. Then they are validated by the domain experts to ensure that the system model representation complies with the specification of the real-world system and the system requirements.

However, through the deployment of SysMM on several projects, the implementation of V&V solutions started to get very challenging due to the many changes triggered from the various dimensions such as the applications of modeling (e.g., functional description and variant management) and the hierarchy levels (e.g., train, consist and subsystems). Therefore, the need for a generic and reusable V&V solution was addressed to improve the V&V activities and hence optimize the deployment of MBSE. The targeted approach was built on the following objectives:

- Enable formal, generic and reusable V&V methods to be used across different projects and different departments.

- Ensure an early start of the V&V activities with regards to the system models development and keep it running in parallel to the SysMM tasks.
- Support V&V automation as much as possible to reduce the time consumed on V&V activities and avoid any potential errors due to manual actions.

3 State of the Art

As explained in [7,8], SysML on its own is not the best suited to apply a development method or build meaningful models in systems engineering. We have to ensure that we manipulate system concepts that are represented by corresponding model elements, along with proper semantic and relationships. A good example would be the lack of elements representing a function, which lead to the creation of specific methods on how to define a functional architecture based on SysML [9]. However, it is possible to adapt SysML to our needs through the use of profile, constraints and additional semantic. The Arcadia method [7] is an example of an adaptation of SysML to system development using system concepts. Arcadia is not considered as a DSML by its creators because of the broad scope of its application and its links to modeling standards. However, Arcadia does not follow the SysML standard fully, and has fixed concepts linked to the modeling elements. BT developed a profile that aims to give semantics to SysML elements while following a general modeling method that could be used also for other systems beside trains. From this comes the need for the verification of the models according to the semantics defined in the profile. The difference with Arcadia is that we can adapt our semantics and our profile depending on ours needs and the method used, without relying on a fixed solution and tool.

We consider here an existing solution for SysML model V&V and several examples of its application. The same way system V&V is different from model V&V, there are differences in the ways to perform V&V. Before considering common V&V solutions such as tests or model checking, which would require for the model to be executable, we want to check if its construction is coherent and holds correct meaning compared to a real system. As shown in [10,11], it is possible to have an implementation and verification of a SysML profile through the use of the Object Constraint Language (OCL) [12]. OCL enables to define constraints on a model, which we refer to as *rules* in this paper. We speak of verification rules and validation rules depending on their usage, but they are often called validation rules in practice, as shown in the several tools using this mechanism [13–15].

While OCL is widely used for this purpose, V&V rules can be developed in other languages supported by the modeling or analysis tools. For this reason, we consider the model V&V solution studied in this paper to be the rules mechanism, whether the rules themselves are coded in OCL or some other language.

Regarding the use of OCL to check or analyze a model, we can find several examples of its adaptation to industrial context and needs, offering technical solutions [16]. Some, such as [17] include OCL as a V&V solution in a process for models and instances design. In this paper, we focus on its use in a much broader context, that is a system development process including many kind of models and taking into account the work of several modeling teams across different projects. We use OCL rules to enforce a semantic and detect error in the model representing the system. Validation using OCL rules is technically possible but it is currently not practical to develop those in a project context, as it will be explained further in this document.

4 BT SysMM V&V

4.1 Method Stakeholders

Figure 2 shows the context of SysMM V&V and the roles of its stakeholders. The V&V activities are part of SysMM and embedded within each task of SysMM (e.g., Operational Analysis). They start in parallel and continue until the deliverables of the SysMM task are verified and validated. Moreover, there is a common V&V part across all the tasks of SysMM, related to the generic and reusable models (such as the model library elements and glossary).

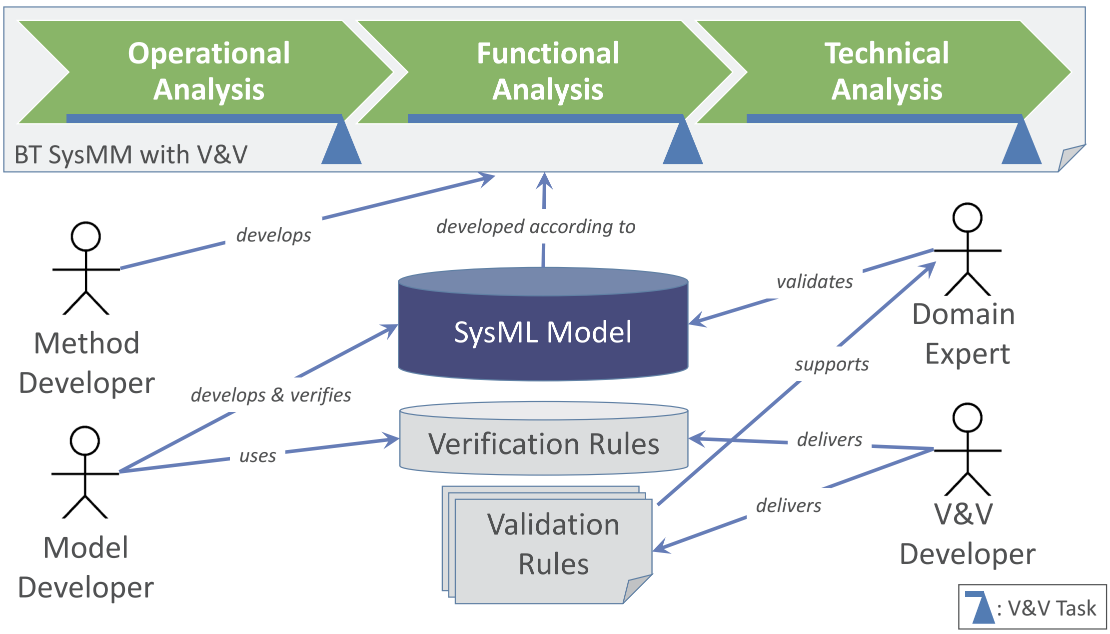

Fig. 2. BT SysMM verification and validation stakeholders context

The context in Fig. 2 indicates that the SysML model is the system of interest under which the V&V takes place. The verification rules are also represented with a model icon because they are implemented using OCL directly in the systems modeling tool. Both the SysML model and the verification rules are included in a project model, whereas the validation rules are documented in a formal textual format and shared through a common guideline. Validation rules are currently broad and/or abstract considerations that cannot be evaluated by a script. While verification check the model and its semantic, validation targets the information expressed in the model regarding the system requirements and expectations. We could define lists of validation rules that check specific considerations expressed by domain experts, but quite often, the lack of resources (time and skills) to develop and use such rules during the project is a challenge.

The SysML model represents an abstraction of the real world system (e.g., train, subsystems or components). Furthermore, the SysML model is being developed based on the defined method and guidelines bundled here with the BT SysMM. The SysMM V&V identifies four stakeholder roles with their own responsibilities and competencies. Table 1 lists these four roles and describes them in detail.

Table 1. The BT SysMM V&V stakeholders roles description

Stakeholder	Role description
Method developer	Is responsible for defining and developing the system modeling method, its guidelines, training courses and tools' customization specifications. This also includes the V&V method parts and their relationship to other method parts. The method developer possesses a unique governance role in monitoring the deployment of the method on projects to ensure the reusability of delivered system models
Model developer	Is a member of the modeling team that is responsible to develop the system models and verify them according to a defined set of verification rules based on project needs. The verification process is done automatically by the system modeling tool and can be set to be active all the time or triggered by the model developer
V&V developer	Is responsible to develop and maintain the verification and validation rules based on the input from the method developer, domain expert and project needs. Additionally, this includes analyzing the V&V requirements, implementing, testing and delivering them. It is the role of the V&V developer to ensure the reusability of V&V rules across several projects
Domain expert	Is a member of the architects team who possess the authority and knowledge in a particular railway technical domain, e.g., brake, propulsion or train control. The domain expert plays a crucial role in validating the system models' content based on his own experience of the real-world system represented by system models

It is crucial for these roles to be well defined in the company, as not everyone should be able to define, develop, apply or change rules implementing the modeling method or defining the conditions that models have to satisfy to be validated. While we can define any arbitrary number of users, the definition of

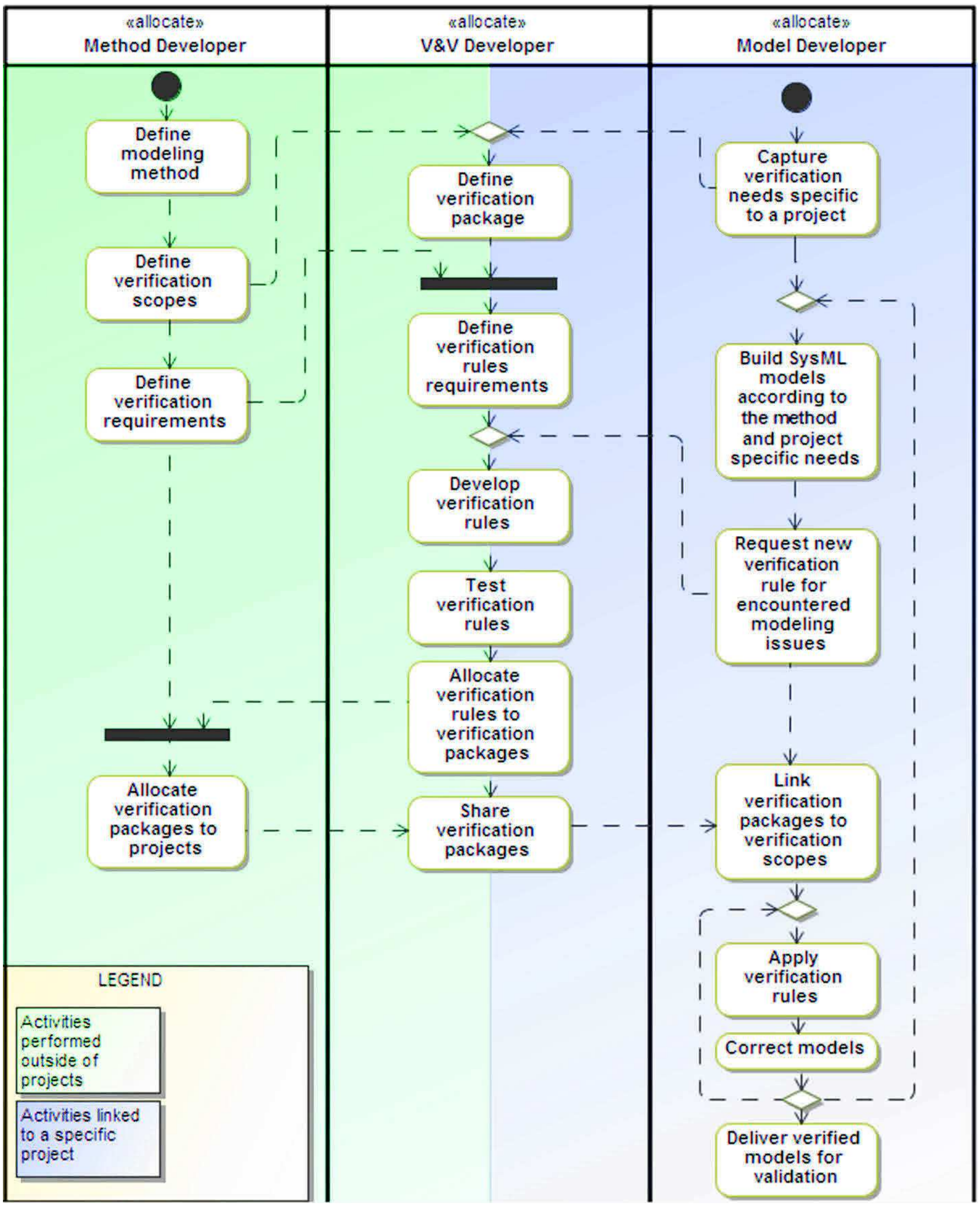

Fig. 3. Verification method

the modeling method and the management of the rules and their packages should be allocated to specific entities. This allows to centralize the skills, development efforts, and rules specifications, while avoiding conflicts and incoherences among the modeling teams. This is part of how we can address the challenges in change management, reusability and conflicting rules discussed later in Sect. 6.

4.2 V&V Method Overview

As we presented the different roles in the previous section, we now show the method and work process they follow in order to specify, develop, share and apply verification rules. This is illustrated in Fig. 3. As validation rules are not yet managed using the rules mechanism, they are not part of the method presented.

Rules are defined and used for a specific purpose and context. A key aspect of our implemented solution is the allocation of verification rules to modules or packages. In this way, rules in a same package can share a same application context and correspond to a same step in the modeling method with the related semantics. As the packages are managed by the method developers, they can be communicated to any team, enabling uniformity and reuse. Rules specific to a project will be contained in their own package. When working on a server, the packages can be automatically updated. Choosing the right verification packages enable us to define, apply and adapt our semantics. Packages can be versioned to be able to work on older projects. We can define packages providing the semantics of other modeling methods when working with or for other providers. Packages are to be built so as to separate conflicting rules.

Verification rules are not just a technical solution, they are specifications on how the modelers should work and what they should deliver. In order to specify, communicate, understand and use the rules, a proper documentation is required. Supposing you work with teams with different tools or an external provider, you can communicate the rules that have to be followed during modeling, even if they are not implemented or compatible with the tools. The documentation should at least specify for each rule: an ID, a target, a method, its current place in the life-cycle and the specification/constraint/error addressed by the rule.

4.3 Benefits

Aside from the semantics, the rules enforce the (modeling) methods and support work processes. Checking the rules on each step results in a report on the quality and level of advancement in the work done, enabling to proceed to the next development step after having checked for errors. Note that by verifying the relationships between concepts/elements, we ensure a certain degree of traceability. Supposing that we have modeled the requirements as artifacts, we can achieve part of the system validation just by ensuring that they are linked to other elements such as functions or scenarios. This is also true across abstraction levels, when switching the SOI from the system to a sub-system.

An advantage of the approach based on verification rules is that it is progressive, empiric, iterative and adaptive. We can specify, update and change the semantics and modeling rules over time. Note that most verification rules should be decided at the start of a project. While we can always develop rules during a project in answer to an immediate need, we should not remove or change any of them once the modeling activities have started. A key point in BT is that new modeling methods are being developed and spread in the different company sites across the world, and with rules they are supported by a common and automatic solution. Modeling teams can check the models and learn at the same time the method implemented by the rules. They also provide a feedback and request new rules. Rules support the training of modeling teams as the rules enforce the way the method has to be applied. In return, the method developers learn from the experience of modeling teams. This create a dynamics that optimizes the work performed and the results obtained across projects, each supplying new rules and improvements. This would not be possible if we were to impose a new tool with a fixed semantics.

4.4 Issues

Before and during development, we came across several issues that had to be solved, such as how to define our rules or how to reduce the time needed for them to be checked. Some of these were problems we wanted to address with our solution, others resulted from its implementation. This enabled us to express new needs and opportunities that will be presented as challenges in Sect. 6. There is also the matter of maturity between verification and validation rules. Finding a way to develop, apply and check validation rules in an automatic way will be the object of further studies. In the rest of the document, we focus on issues related specifically to verification rules or ones that apply to the rule mechanism solution as a whole.

5 Use Case Example

Traditionally, the work split during the model development between teams is based on the work breakdown structure which defines a list of scopes covering all functionalities of the SOI (e.g., train or subsystem). The functional scope *travel direction*, taken from [2], is used in this section to illustrate the application of SysMM V&V on an example from the railway domain. A scope here is referred to a part of the work breakdown structure of the whole function set. Figure 4 shows some of the SysML diagrams delivered using the SysMM operational and functional analysis tasks.

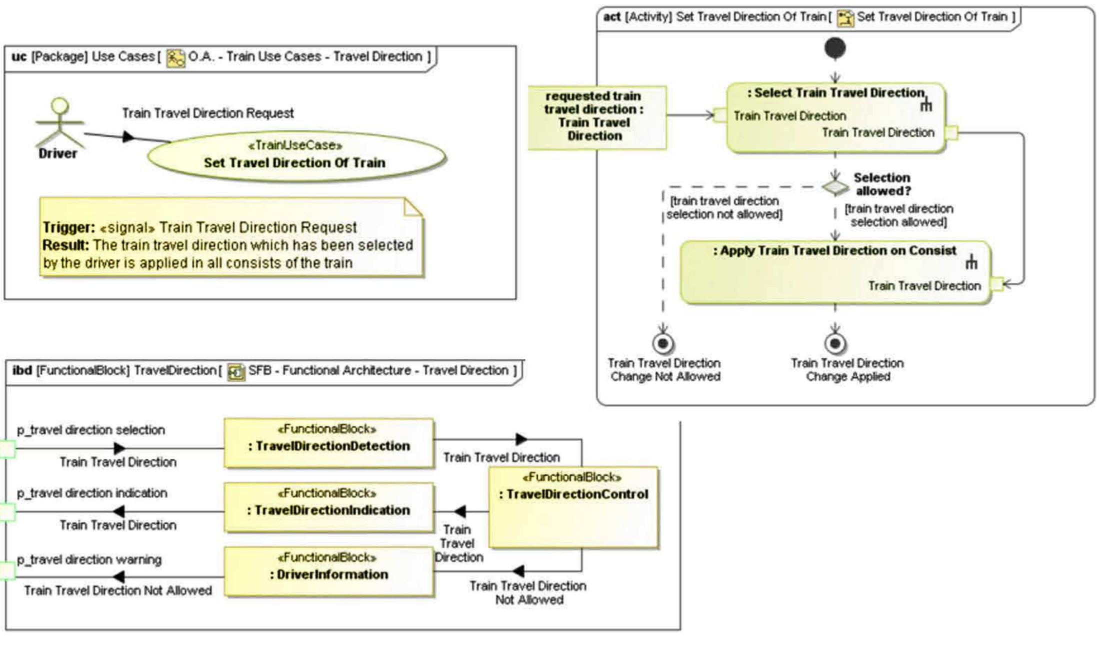

Fig. 4. BT SysMM diagrams example on which V&V is applied [2]

The operational analysis part is demonstrated through the use case and activity diagrams. The use case diagram defines the use case *"Set Travel Direction Of Train"*, its actor (i.e., the driver) and the respective trigger signal *"Train Travel Direction Request"*. The activity diagram describes the internal behavior of this use case in a generic manner independent from any specific functional or technical solution in order to reuse it in several projects.

The functional analysis part is shown with the internal block definition diagram where previously modeled activities are structured in a functional architecture that fits a particular train platform or project. The functional architecture defines all functions needed to cover the *travel direction* scope with functional blocks and their interfaces. These function blocks are linked back to the activities of the operational analysis and allocated later on to the technical blocks solution to ensure traceability.

The scope discussed here is one out of other hundreds of scopes normally modeled to describe the safety related functions of a train. Usually, a set of scopes is assigned to particular domain experts and model development team. The model developers, usually system engineers, takes the responsibility to develop the SysML models based on the input requirements of their own scope.

During the modeling activities, the model developers *verify* their models automatically based on the verification rules implemented in the tool. These rules are aligned with the deployed method and implemented using OCL in the systems modeling tool. Table 2 lists a sample of textual representation of the verification rules for model elements such as use cases or signals. The verification rules check automatically if model elements are modeled according to the defined

Table 2. BT SysMM rules examples

Sample Verification Rules:	
1.	A use case must own at least one activity
2.	A use case name must follow the naming convention guidelines (e.g., starting with a verb and all words are capitalized)
3.	A triggered use case must have at least one actor and one trigger signal
4.	A signal name must follow the naming convention guidelines
5.	Model elements, e.g., use cases, must be unique across the whole model
6.	Each function is linked to at least one activity
Sample Validation Rules:	
1.	Are the use cases' actors complete according to the requirements?
2.	Are all actors and signals considered in the correct way with respect to the requirements linked to the use case?
3.	Does the use case activity describe the exact scenario of real operation as described in the requirements?
4.	Is the functional architecture solution (i.e., functional split and allocation) satisfying the relevant requirements?

method. If not, the model developer is getting a notice about the result of the check i.e., an error, warning or information. One can see from the list that the verification rules check also model consistency and completion.

After the model is verified, it is shared with the responsible domain expert for the sake of *validation*. The second part of Table 2 lists a sample of the validation rules relevant to the presented example. These rules are documented in a formalized textual format and offered to support the domain expert during his model validation activities. The validation rules are always traced back to the system requirements. It is the role of the domain expert to apply his experience in order to check these traceability links and confirm that the model specification is valid with respect to the provided requirements.

6 V&V Adoption Challenges

In the following, we contribute by providing the list of challenges faced during the adoption of the V&V work on SysML models at BT and discuss the need to achieve a common interpretation of them in order to start solving them.

SysML Tools Integration. It is often the situation within large organizations that different departments or different sites' locations use different SysML tools. Challenges do not lie in the tool's diversity instead with the integration between the different tools. Model exchange (i.e., elements and diagrams) is still hardly possible between different SysML tools. In case tool vendors offer it (i.e., normally with the XMI file exchange), an additional effort and customization is always required. The early phase objective of BT was to achieve a model V&V solution across different SysML tools from the beginning of the system model development and not only at the end, i.e., model delivery from each tool. Unfortunately, this objective for having a common SysML V&V solution across different SysML tools is still not possible without having an extra SysML-independent V&V tool which is not preferred. This lead us to have a flexible solution that is the rule mechanism. Rules can be documented and transmitted even if they have different implementations. What is lacking is a way to quickly develop, deploy and/or adapt them across tools and teams. As the rules relate to concepts and data, part of the solution could be to apply them on a metamodel and transport them from one tool to another, based on the work with OSLC [18,19].

Complexity with Large SysML Models. The evolution of systems, components and functions has hugely contributed in growing the number of elements and relations of SysML models. For instance, modeling one functional scope, e.g., brake control or propulsion of a railway vehicle on the vehicle level only, includes in average 20 use cases, 150 signals and 25 function blocks. Covering only the safety related functions on the vehicle level, one should multiply these numbers by around 100 other scopes. The issue here is not only with the high number of model elements but with the dependencies inside a model or across several models. Very often when dealing with large systems and large teams, complexity issues arise where existing tools and methods can reach their limits in solving them. This kind of high complexity levels has a huge influence on the V&V benefit. Therefore, a suitable solution should deal with the complexity issues and not only on tools-level but also on methods and processes.

Conflicting V&V Rules. As we do not rely on a tool applying a DSML and separating the different scopes of study, we tend to use the same SysML elements for different purposes. For example, while we do not apply a stereotype on a sequence diagram, we do not use the same type of signals in the messages depending on whether we are making an operational analysis, that is an analysis of the system behavior from an external point of view, or whether we are making a functional analysis, meaning we specify the signals exchange between functions. Using the right signals according to one analysis will result in not using those required for the other. Applying a verification rule depends on the verification goal, the scope and the type of the element. If we check all verification rules on the whole project, then independently from performance issues, there will be contradictions and the model will never be considered correct.

V&V Managed Reuse. Many organizations still follow an opportunistic and isolated reuse approach, where a set of data is copied and pasted from one context to another. Unfortunately, this still happens even with work performed on SysML models and results in losing the "source of truth" as soon as the copied source or pasted target is changed. During the early stages of modeling, V&V rules are often created only for a particular deployment (project or product) and thus specific. Reusability between different deployments gets complicated without proper modularity concepts for defining the communality and variability of the V&V rules. According to BT MBSE objectives, reuse is a key factor for improving the system models V&V efficiency. In order to achieve a managed V&V reuse, modularity, governance and variability management must be in place for V&V solutions.

V&V Change Management. The work with V&V rules is subject to many changes, most of which are due to models or rules modifications, emerging from different stakeholders for the aim of optimization – of V&V rules. Working around those changes is eased by following our method, managing rules in PLM and documenting them, but it is not enough. Each change request triggers a sequence of tasks (e.g., review request with impact analysis, change approval, change implementation with review and reporting) in order to reach the final successful implementation and closing the change request. Such tasks are often grouped under the term of change management. Although the usage of methods as agile, scrum and kanban helps a lot in addressing the change and delivering value with a quick impact and continual basis, the responsibility still lies on the personnel side of the team involved. Particularly, dependencies between change requests are often not visible from the beginning, consume time and impact the V&V solution delivery timeline. Moreover, the integration between agile tools and SysML tools – to achieve a full traceability – is still very challenging when dealing with multi-user environment. Although the technology is heading towards cloud based solutions, their low performance due to large models is still an issue.

V&V Optimization. Although the work done at BT with regard to V&V has evolved enormously for the aim of optimization during the last years, it still requires high effort to analyze all relevant rules and the order of executing them to deliver better V&V results. Particularly, defining the dependencies between the verification rules is still very challenging and needs a lot of tool customizations, specific method solutions and personnel effort. Therefore, there is a need to investigate which other domains could solve the optimization challenges, for instance in [20] the combination of Statistical Machine Learning and OCL demonstrates how Artificial Intelligence can support in solving this challenge and in [21] an implementation of machine learning for a model-based conceptual design evaluation is demonstrated.

7 Conclusion and Future Directions

In this paper we presented the work related to the verification and validation of SysML models from an industrial perspective. Although the usage of OCL is well known for model verification, we first contributed by describing the method and roles used at BT to achieve efficient V&V results and accelerate the system development process with less time consumed on testing and system validation. Our second contribution concerned the description of common challenges faced with model-based V&V in large organizations. After having identified these challenges, a common understanding between the whole modeling team was achieved to justify the reasons behind the previous pitfalls and failures.

Our future work spans in two main directions: on one hand, we aim to analyze and the describe the fulfillment of the V&V method developed in relation to the challenges discussed in this paper. In so doing, we expect to identify the challenges which cannot be solved through method, process or tool solutions. On the other hand, we aim to apply new domains, such as Artificial Intelligence (AI) and machine learning to use the large amount of available data, let the AI system learn from it and support with an optimized V&V results. Finally, we aim to use this work to trigger the MBSE community and particularly the SysML working group in upcoming conference workshops to consider V&V more in detail in future SysML versions (e.g., 2.0) in order to solve industrial adoption challenges from a language prescriptive. The SysML V2 working group [22] states that the next version of SysML should enable a concise representation of the concepts and be able to validate that the model is logically consistent. It should also be highly adaptable and customizable in regard of domain specific concepts. The rules mechanism presented here enables to do both, and it would be interesting to be able to express the rules based on the SysML language rather than on its implementation in tools, while taking in account the other challenges we expressed.

Acknowledgements. This work is supported by Bombardier Transportation SAS and the ANRT CIFRE grant #2016/0262.

References

1. OMG: OMG Systems Modeling LanguageTM Version 1.5 (2017). http://www.omg.org/spec/SysML/1.5/
2. Chami, M., Oggier, P., Naas, O., Heinz, M.: Real world application of MBSE at Bombardier Transportation. In: The Swiss Systems Engineering Day (SWISSED 2015), Kongresshaus Zurich, 8th September 2015
3. INCOSE: Systems Engineering Vision 2020. Version 2.03 edn. Technical Operations, International Council on Systems Engineering (INCOSE), September 2007. INCOSE-TP-2004-004-02
4. IEEE: 1012–2012 IEEE Standard for System and Software Verification and Validation
5. Stecklein, J.M., Dabney, J., Dick, B., Haskins, B., Lovell, R., Moroney, G.: Error cost escalation through the project life cycle (2004)
6. Chami, M., Morkevicius, A., Aleksandraviciene, A., Bruel, J.M.: Towards solving MBSE adoption challenges: the D3 MBSE adoption toolbox. In: 28th Annual INCOSE International Symposium, Washington DC, USA, 7–12 July (2018)
7. Bonnet, S., Exertier, D., Normand, V.: Not (strictly) relying on SysML for MBSE: language, tooling and development perspectives (2015). OCLC: 255348295
8. Guizzardi, G.: Ontological foundations for structural conceptual models. Ph.D. thesis, Centre for Telematics and Information Technology, Telematica Instituut, Enschede, The Netherlands (2005)
9. Lamm, J.G., Weilkiens, T.: Functional Architectures in SysML. In: Proceedings of the Tag des Systems Engineering (TdSE 2010), Munich, Germany (2010)
10. Berkenkotter, K.: OCL-Based Validation of a Railway Domain Profile (2006). OCLC: 248918751
11. Dragomir, I., Ober, I., Percebois, C.: Contract-based modeling and verification of timed safety requirements within SysML. Softw. Syst. Model. **16**(2), 587–624 (2017)
12. OMG Specification: Object Constraint Language V.2.4. Object Management Group pct/07-08-04, February 2014
13. Eclipse - Papyrus Guide: Validate (OCL) constraints of a profile
14. Sparx Systems: Model Validation (2016)
15. No Magic Documentation: Creating validation rules - MagicDraw 18.2
16. Altenhofen, M., Hettel, T., Kusterer, S.: OCL support in an industrial environment (2006)
17. Delmas, R., Pires, A.F., Polacsek, T.: A verification and validation process for model-driven engineering. EDP Sciences 455–468 (2013)
18. OMG: OSLC4mbse Working Group
19. Shani, U.: A Case for a SysML OWL Ontology (2014)

20. Zhao, L., Li, F.: Statistical machine learning in natural language understanding: object constraint language translator for business process. In: 2008 IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, pp. 1056–1059, December 2008
21. Chami, M., Bruel, J.: Towards an integrated conceptual design evaluation of mechatronic systems: the SysDICE approach. In: Proceedings of the International Conference on Computational Science, Reykjavík, Iceland, pp. 650–659 (2015)
22. OMG: SysML v2 RFP Working Group